Lecture 3:

ATmega16 Ports, Clock Sources

status register

- **status register** (SREG)
- It is 8-bit long each bit has a different meaning.

Bit	7	6	5	4	3	2	1	0	
	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

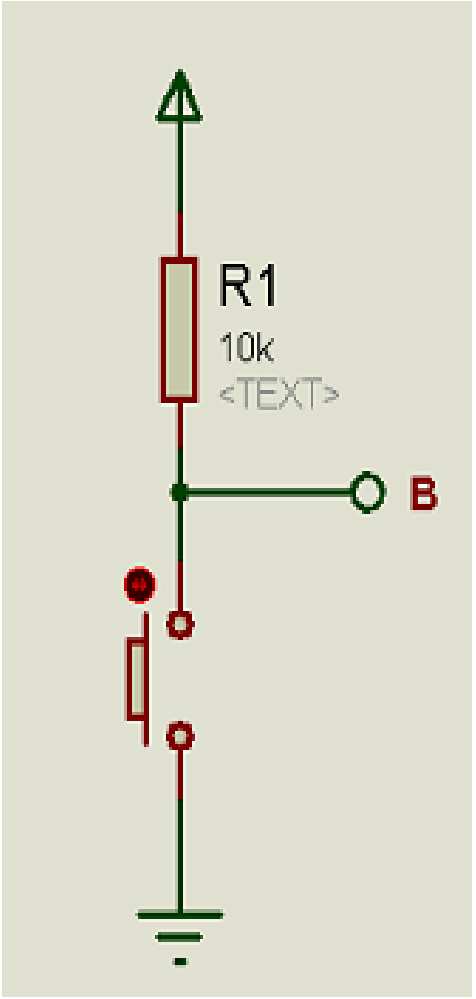
- I: Global Interrupt Enable/Disable Flag, SREG7
- T: Transfer bit used by BLD and BST instructions, SREG6
- H: Half Carry Flag, SREG5
- S: N ⊕ V, For signed tests, SREG4
- V: Two's complement overflow indicator, SREG3
- N: Negative Flag, SREG2
- Z: Zero Flag, SREG1
- C: Carry Flag, SREG0

Port A (PA7..PA0)

- Port A serves as the analog inputs to the A/D Converter.
- Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used.
- Port pins can provide internal pull-up resistors (selected for each bit).
- The Port A output buffers have symmetrical drive characteristics with both high sink and source capability.
- When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated.
- The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Pull Up Resistor

- We use it to prevent the pin’s voltage to be floating due to the noises .
- When the switch is pressed , we get input LOW on the pin “B”.
- When the switch isn’t pressed , we get input HIGH on the pin “B”



Port A Pins Alternante Fonctions

Port Pin	Alternate Function
PA7	ADC7 (ADC input channel 7)
PA6	ADC6 (ADC input channel 6)
PA5	ADC5 (ADC input channel 5)
PA4	ADC4 (ADC input channel 4)
PA3	ADC3 (ADC input channel 3)
PA2	ADC2 (ADC input channel 2)
PA1	ADC1 (ADC input channel 1)
PA0	ADC0 (ADC input channel 0)

Port B (PB7..PB0)

- Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit).
- The Port B output buffers have symmetrical drive characteristics with both high sink and source capability.
- As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated.
- The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.
- Port B also serves the functions of various special features…..

Port B Pins Alternante Fonctions

Port Pin	Alternate Functions
PB7	SCK (SPI Bus Serial Clock)
PB6	MISO (SPI Bus Master Input/Slave Output)
PB5	MOSI (SPI Bus Master Output/Slave Input)
PB4	$\overline{SS}$ (SPI Slave Select Input)
PB3	AIN1 (Analog Comparator Negative Input) OC0 (Timer/Counter0 Output Compare Match Output)
PB2	AIN0 (Analog Comparator Positive Input) INT2 (External Interrupt 2 Input)
PB1	T1 (Timer/Counter1 External Counter Input)
PB0	T0 (Timer/Counter0 External Counter Input) XCK (USART External Clock Input/Output)

Port C (PC7..PC0)

- Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit).
- The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated.
- The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs.
- Port C also serves the functions of the JTAG interface and other special features …..

Port C Pins Alternante Fonctions

Port Pin	Alternate Function
PC7	TOSC2 (Timer Oscillator Pin 2)
PC6	TOSC1 (Timer Oscillator Pin 1)
PC5	TDI (JTAG Test Data In)
PC4	TDO (JTAG Test Data Out)
PC3	TMS (JTAG Test Mode Select)
PC2	TCK (JTAG Test Clock)
PC1	SDA (Two-wire Serial Bus Data Input/Output Line)
PC0	SCL (Two-wire Serial Bus Clock Line)

Port D (PD7..PD0)

- Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit).
- The Port D output buffers have symmetrical drive characteristics with both high sink and source capability.
- As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated.
- The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.
- Port D also serves the functions of various special features…

Port D Pins Alternante Fonctions

Port Pin	Alternate Function
PD7	OC2 (Timer/Counter2 Output Compare Match Output)
PD6	ICP1 (Timer/Counter1 Input Capture Pin)
PD5	OC1A (Timer/Counter1 Output Compare A Match Output)
PD4	OC1B (Timer/Counter1 Output Compare B Match Output)
PD3	INT1 (External Interrupt 1 Input)
PD2	INT0 (External Interrupt 0 Input)
PD1	TXD (USART Output Pin)
PD0	RXD (USART Input Pin)

Clock Sources

Device Clocking Option	CKSEL3..0
External Crystal/Ceramic Resonator	1111 - 1010
External Low-frequency Crystal	1001
External RC Oscillator	1000 - 0101
Calibrated Internal RC Oscillator	0100 - 0001
External Clock	0000

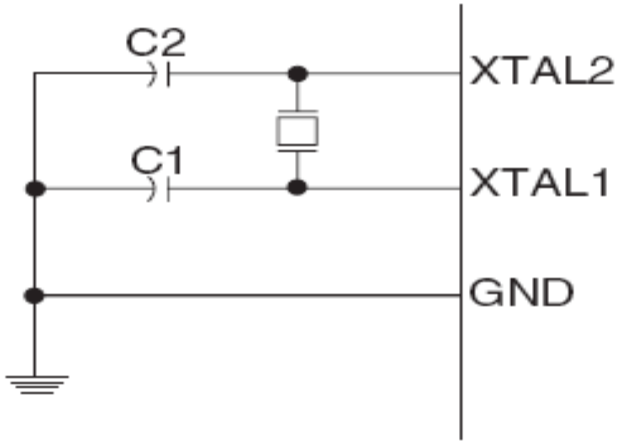
Note: 1. For all fuses “1” means unprogrammed while “0” means programmed.

The device is shipped with CKSEL = “0001” and SUT = “10”. The default clock source setting is therefore the
1 MHz Internal RC Oscillator with longest startup time

Crystal Oscillator

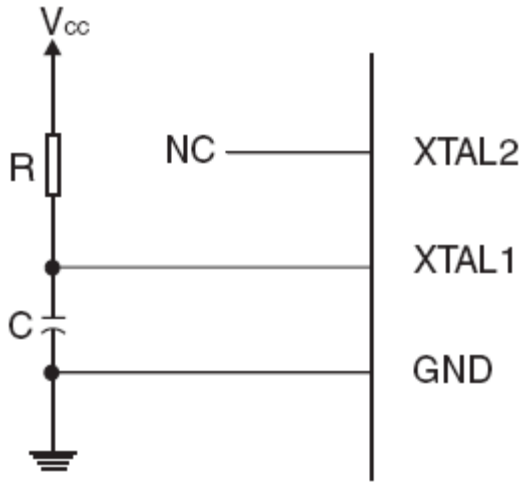
CKOPT	CKSEL3..1	Frequency Range (MHz)	Recommended Range for Capacitors C1 and C2 for Use with Crystals (pF)
1	101[1]	0.4 - 0.9	–
1	110	0.9 - 3.0	12 - 22
1	111	3.0 - 8.0	12 - 22
0	101, 110, 111	1.0 ≤	12 - 22

Note: 1. This option should not be used with crystals, only with ceramic resonators.



External RC Oscillator

CKSEL3..0	Frequency Range (MHz)
0101	0.1 ≤ 0.9
0110	0.9 - 3.0
0111	3.0 - 8.0
1000	8.0 - 12.0



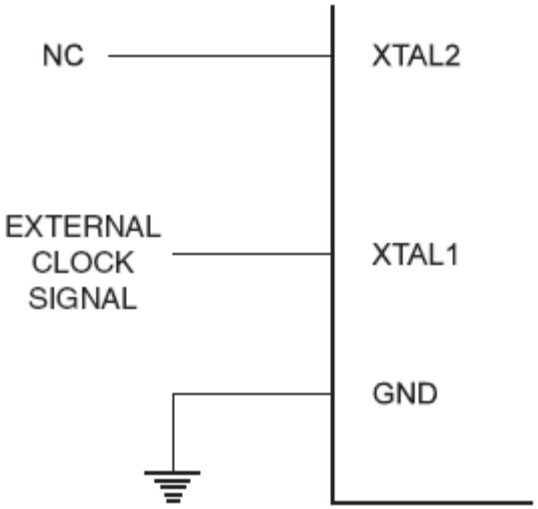
Calibrated Internal RC Oscillator

CKSEL3..0	Nominal Frequency (MHz)
0001[1]	1.0
0010	2.0
0011	4.0
0100	8.0

Note: 1. The device is shipped with this option selected.

External Clock

To run the device on an external clock, the CKSEL Fuses must be programmed to “0000”.



ATmega16 Instruction Set

Instruction Set Nomenclature: Registers and Operands

Registers and Operands

Rd:	Destination (and source) register in the Register File
Rr:	Source register in the Register File
R:	Result after instruction is executed
K:	Constant data
k:	Constant address
b:	Bit in the Register File or I/O Register (3-bit)
s:	Bit in the Status Register (3-bit)
X,Y,Z:	Indirect Address Register (X=R27:R26, Y=R29:R28 and Z=R31:R30)
A:	I/O location address
q:	Displacement for direct addressing (6-bit)

Instruction Set Summary

Mnemonics	Operands	Description	Operation	Flags	#Clock Note
Arithmetic and Logic Instructions					
ADD	Rd, Rr	Add without Carry	Rd ← Rd + Rr	Z,C,N,V,S,H	1
ADC	Rd, Rr	Add with Carry	Rd ← Rd + Rr + C	Z,C,N,V,S,H	1
ADIW	Rd, K	Add Immediate to Word	Rd+1:Rd ← Rd+1:Rd + K	Z,C,N,V,S	2 (1)
SUB	Rd, Rr	Subtract without Carry	Rd ← Rd - Rr	Z,C,N,V,S,H	1
SUBI	Rd, K	Subtract Immediate	Rd ← Rd - K	Z,C,N,V,S,H	1
SBC	Rd, Rr	Subtract with Carry	Rd ← Rd - Rr - C	Z,C,N,V,S,H	1
SBCI	Rd, K	Subtract Immediate with Carry	Rd ← Rd - K - C	Z,C,N,V,S,H	1
SBIW	Rd, K	Subtract Immediate from Word	Rd+1:Rd ← Rd+1:Rd - K	Z,C,N,V,S	2 (1)
AND	Rd, Rr	Logical AND	Rd ← Rd • Rr	Z,N,V,S	1
ANDI	Rd, K	Logical AND with Immediate	Rd ← Rd • K	Z,N,V,S	1
OR	Rd, Rr	Logical OR	Rd ← Rd v Rr	Z,N,V,S	1
ORI	Rd, K	Logical OR with Immediate	Rd ← Rd v K	Z,N,V,S	1
EOR	Rd, Rr	Exclusive OR	Rd ← Rd ⊕ Rr	Z,N,V,S	1
COM	Rd	One's Complement	Rd ← $FF - Rd	Z,C,N,V,S	1
NEG	Rd	Two's Complement	Rd ← $00 - Rd	Z,C,N,V,S,H	1
SBR	Rd,K	Set Bit(s) in Register	Rd ← Rd v K	Z,N,V,S	1
CBR	Rd,K	Clear Bit(s) in Register	Rd ← Rd • ($FFh - K)	Z,N,V,S	1

ADD – Add without Carry

Description:

Adds two registers without the C Flag and places the result in the destination register Rd.

Operation:

(i) Rd ← Rd + Rr

	Syntax:	Operands:	Program Counter:
(i)	ADD Rd,Rr	$0 \le d \le 31, 0 \le r \le 31$	PC ← PC + 1

16-bit Opcode:

0000	11rd	dddd	rrrr

Status Register (SREG) and Boolean Formula:

I	T	H	S	V	N	Z	C
–	–	⇔	⇔	⇔	⇔	⇔	⇔

Instructions with similar Syntax as ADD

- ADC Rd, Rr
- SUB Rd, Rr
- SBC Rd, Rr
- AND Rd, Rr
- OR Rd, Rr
- EOR Rd, Rr
- CP Rd, Rr

Example:

```
add   r1,r2     ; Add r2 to r1 (r1=r1+r2)

add   r28,r28   ; Add r28 to itself (r28=r28+r28)
```

ADIW – Add Immediate to Word

Description:

Adds an immediate value (0 - 63) to a register pair and places the result in the register pair. T
upper four register pairs, and is well suited for operations on the pointer registers.

This instruction is not available in all devices. Refer to the device specific instruction set sum

Operation:

(i) Rd+1:Rd ← Rd+1:Rd + K

	Syntax:	Operands:	Program Counter:
(i)	ADIW Rd+1:Rd,K	d ∈ {24,26,28,30}, 0 ≤ K ≤ 63	PC ← PC + 1

16-bit Opcode:

1001	0110	KKdd	KKKK

Status Register (SREG) and Boolean Formula:

I	T	H	S	V	N	Z	C
–	–	–	⇔	⇔	⇔	⇔	⇔

Instructions with similar Syntax as ADIW

- ADIW Rd, K
- SBIW Rd, K

Example:

```
adiw  r25:24,1  ; Add 1 to r25:r24
adiw  ZH:ZL,63  ; Add 63 to the Z-pointer(r31:r30)
```

ANDI – Logical AND with Immediate

Description:

Performs the logical AND between the contents of register Rd and a constant and places the
ter Rd.

Operation:

(i) $Rd \leftarrow Rd \bullet K$

	Syntax:	Operands:	Program Counter:
(i)	ANDI Rd,K	$16 \le d \le 31, 0 \le K \le 255$	$PC \leftarrow PC + 1$

16-bit Opcode:

0111	KKKK	dddd	KKKK

Status Register (SREG) and Boolean Formula:

I	T	H	S	V	N	Z	C
–	–	–	⇔	0	⇔	⇔	–

Instructions with similar Syntax as ANDI

- ANDI Rd, K
- SUBI Rd, K
- SBCI Rd, K
- ORI Rd, K

- COM Rd
- NEG Rd
- INC Rd
- DEC Rd
-

Example:

```
andi  r17,$0F   ; Clear upper nibble of r17
andi  r18,$10   ; Isolate bit 4 in r18
andi  r19,$AA   ; Clear odd bits of r19
```

MOV – Copy Register

Description:

This instruction makes a copy of one register into another. The source register Rr is left unc
register Rd is loaded with a copy of Rr.

Operation:

(i) Rd ← Rr

	Syntax:	Operands:	Program Counter:
(i)	MOV Rd,Rr	$0 \leq d \leq 31, 0 \leq r \leq 31$	PC ← PC + 1

16-bit Opcode:

0010	11rd	dddd	rrrr

MOVW – Copy Register Word

Description:

This instruction makes a copy of one register pair into another register pair. The source
unchanged, while the destination register pair Rd+1:Rd is loaded with a copy of Rr + 1:Rr.

This instruction is not available in all devices. Refer to the device specific instruction set sum

Operation:

(i) Rd+1:Rd ← Rr+1:Rr

	Syntax:	Operands:	Program Counter:
(i)	MOVW Rd+1:Rd,Rr+1	Rd ∈ {0,2,...,30}, r ∈ {0,2,...,30}	PC ← PC + 1

16-bit Opcode:

0000	0001	dddd	rrrr

Example:

```
        movw    r17:16,r1:r0 ; Copy r1:r0 to r17:r16
        call    check        ; Call subroutine
        ...
check:  cpi     r16,$11      ; Compare r16 to $11
        ...
        cpi     r17,$32      ; Compare r17 to $32
        ...
        ret                  ; Return from subroutine
```

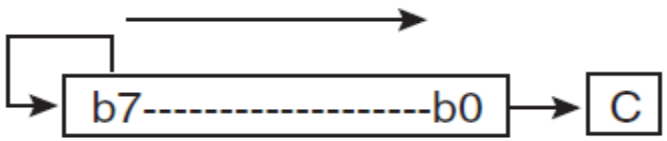
ASR – Arithmetic Shift Right

Description:

Shifts all bits in Rd one place to the right. Bit 7 is held constant. Bit 0 is loaded into the C Fla
effectively divides a signed value by two without changing its sign. The Carry Flag can be us

Operation:

(i)



	Syntax:	Operands:	Program Counter:
(i)	ASR Rd	$0 \le d \le 31$	PC ← PC + 1

16-bit Opcode:

1001	010d	dddd	0101

Status Register (SREG) and Boolean Formula:

I	T	H	S	V	N	Z	C
–	–	–	⇔	⇔	⇔	⇔	⇔

Instructions with similar Syntax as ASR

- ASR Rd
- LSL Rd
- LSR Rd
- ROL Rd
- ROR Rd
- CPI Rd, K
- SBR Rd, K
- CBR Rd, K

BCLR – Bit Clear in SREG

Description:

Clears a single Flag in SREG.

	Operation:
(i)	SREG(s) ← 0

	Syntax:	Operands:	Program Counter:
(i)	BCLR s	$0 \leq s \leq 7$	PC ← PC + 1

16-bit Opcode:

1001	0100	1sss	1000

Status Register (SREG) and Boolean Formula:

I	T	H	S	V	N	Z	C
⇔	⇔	⇔	⇔	⇔	⇔	⇔	⇔

Instructions with similar Syntax as BCLR

- BCLR s
- BSET s

Example:

```
bclr    0       ; Clear Carry Flag
bclr    7       ; Disable interrupts
```

BLD – Bit Load from the T Flag in SREG to a Bit in Register

Description:

Copies the T Flag in the SREG (Status Register) to bit b in register Rd.

Operation:

(i) Rd(b) ← T

	Syntax:	Operands:	Program Counter:
(i)	BLD Rd,b	$0 \le d \le 31, 0 \le b \le 7$	PC ← PC + 1

16 bit Opcode:

1111	100d	dddd	0bbb

Status Register (SREG) and Boolean Formula:

I	T	H	S	V	N	Z	C
–	–	–	–	–	–	–	–

Instructions with similar Syntax as BLD

- BLD Rd, b
- BST Rr, b

Example:

```
                    ; Copy bit
    bst   r1,2      ; Store bit 2 of r1 in T Flag
    bld   r0,4      ; Load T Flag into bit 4 of r0
```

LD – Load Indirect from Data Space to Register

Using the X-pointer:

	Operation:		Comment:
(i)	Rd ← (X)		X: Unchanged
(ii)	Rd ← (X)	X ← X + 1	X: Post incremented
(iii)	X ← X - 1	Rd ← (X)	X: Pre decremented

Using the Y-pointer:

	Operation:		Comment:
(i)	Rd ← (Y)		Y: Unchanged
(ii)	Rd ← (Y)	Y ← Y + 1	Y: Post incremented
(iii)	Y ← Y - 1	Rd ← (Y)	Y: Pre decremented
(iiii)	Rd ← (Y+q)		Y: Unchanged, q: Displaceme

Using the Z-pointer:

	Operation:	Comment:	
(i)	Rd ← (Z)		Z: Unchanged
(ii)	Rd ← (Z)	Z ← Z + 1	Z: Post increment
(iii)	Z ← Z -1	Rd ← (Z)	Z: Pre decrement
(iiii)	Rd ← (Z+q)		Z: Unchanged, q: Displacem

Example:

```
clr   r27           ; Clear X high byte
ldi   r26,$60       ; Set X low byte to $60
ld    r0,X+     ; Load r0 with data space loc. $60(X post inc)
ld    r1,X      ; Load r1 with data space loc. $61
ldi   r26,$63   ; Set X low byte to $63
ld    r2,X      ; Load r2 with data space loc. $63
ld    r3,-X     ; Load r3 with data space loc. $62(X pre dec)
```

Example:

```
clr   r31      ; Clear Z high byte
ldi   r30,$60  ; Set Z low byte to $60
ld    r0,Z+    ; Load r0 with data space loc. $60(Z post inc)
ld    r1,Z     ; Load r1 with data space loc. $61
ldi   r30,$63  ; Set Z low byte to $63
ld    r2,Z     ; Load r2 with data space loc. $63
ld    r3,-Z    ; Load r3 with data space loc. $62(Z pre dec)
ldd   r4,Z+2   ; Load r4 with data space loc. $64
```

BRBC – Branch if Bit in SREG is Cleared

Description:

Conditional relative branch. Tests a single bit in SREG and branches relatively to PC if the
branches relatively to PC in either direction (PC - 63 ≤ destination ≤ PC + 64). The parameter
represented in two's complement form.

Operation:

(i) If SREG(s) = 0 then PC ← PC + k + 1, else PC ← PC + 1

	Syntax:	Operands:	Program Counter:
(i)	BRBC s,k	0 ≤ s ≤ 7, -64 ≤ k ≤ +63	PC ← PC + k + 1 PC ← PC + 1, if condition is

16-bit Opcode:

1111	01kk	kkkk	ksss

Status Register (SREG) and Boolean Formula:

I	T	H	S	V	N	Z	C
–	–	–	–	–	–	–	–

Instructions with similar Syntax as BRBC

- BRBC s, k
- BRBS s, k

Example:

```
cpi   r20,5     ; Compare r20 to the value 5
brbc  1,noteq   ; Branch if Zero Flag cleared
...
noteq:nop       ; Branch destination (do nothing)
```

BRCC – Branch if Carry Cleared

Description:

Conditional relative branch. Tests the Carry Flag (C) and branches relatively to PC if C is clea
relatively to PC in either direction (PC - 63 ≤ destination ≤ PC + 64). The parameter k is the
sented in two's complement form. (Equivalent to instruction BRBC 0,k).

Operation:

(i) If C = 0 then PC ← PC + k + 1, else PC ← PC + 1

	Syntax:	Operands:	Program Counter:
(i)	BRCC k	-64 ≤ k ≤ +63	PC ← PC + k + 1 PC ← PC + 1, if condition i

16-bit Opcode:

1111	01kk	kkkk	k000

Instructions with similar Syntax as BRCC

1. BRCC k
2. BRCS k
3. BRTS k
4. BRTC k
5. BRVS k
6. BRVC k
7. BRHS k
8. BRHC k
9. BRIE k
10. BRID k
11. BREQ k
12. BRNE k
13. BRSH k
14. BRLO k
15. BRMI k
16. BRPL k
17. BRGE k
18. BRLT k

Example:

```
            add   r22,r23     ; Add r23 to r22
            brcc  nocarry     ; Branch if carry cleared
            ...
nocarry:    nop               ; Branch destination (do nothing)
```

CALL – Long Call to a Subroutine

Description:

Calls to a subroutine within the entire Program memory. The return address (to the instruction after the CALL) will be stor onto the Stack. (See also RCALL). The Stack Pointer uses a post-decrement scheme during CALL.

This instruction is not available in all devices. Refer to the device specific instruction set summary.

Operation:

(i) PC ← k Devices with 16 bits PC, 128K bytes Program memory maximum.
(ii) PC ← k Devices with 22 bits PC, 8M bytes Program memory maximum.

	Syntax:	Operands:	Program Counter	Stack:
(i)	CALL k	$0 \le k < 64K$	PC ← k	STACK ← PC+2 SP ← SP-2, (2 bytes, 16 bits)
(ii)	CALL k	$0 \le k < 4M$	PC ← k	STACK ← PC+2 SP ← SP-3 (3 bytes, 22 bits)

32-bit Opcode:

1001	010k	kkkk	111k
kkkk	kkkk	kkkk	kkkk

Example:

```
        mov     r16,r0          ; Copy r0 to r16
        call    check           ; Call subroutine
        nop                     ; Continue (do nothing)
        ...
check:  cpi     r16,$42         ; Check if r16 has a special value
        breq    error           ; Branch if equal
        ret                     ; Return from subroutine
        ...
error:  rjmp    error           ; Infinite loop
```

CBI – Clear Bit in I/O Register

Description:

Clears a specified bit in an I/O Register. This instruction operates on the lower 32 I/O Regist

Operation:

(i) I/O(A,b) ← 0

	Syntax:	Operands:	Program Counter:
(i)	CBI A,b	$0 \leq A \leq 31, 0 \leq b \leq 7$	PC ← PC + 1

16-bit Opcode:

1001	1000	AAAA	Abbb

Instructions with similar Syntax as CBI

- CBI A, b
- SBI A, b

Example:

```
cbi     $12,7           ; Clear bit 7 in Port D
```

CBR – Clear Bits in Register

Description:

Clears the specified bits in register Rd. Performs the logical AND between the contents of re
of the constant mask K. The result will be placed in register Rd.

Operation:

(i) Rd ← Rd • ($FF - K)

	Syntax:	Operands:	Program Counter:
(i)	CBR Rd,K	$16 \le d \le 31, 0 \le K \le 255$	PC ← PC + 1

16-bit Opcode: (see ANDI with K complemented)

Status Register (SREG) and Boolean Formula:

I	T	H	S	V	N	Z	C
–	–	–	⇔	0	⇔	⇔	–

Example:

```
cbr     r16,$F0   ; Clear upper nibble of r16
cbr     r18,1     ; Clear bit 0 in r18
```

CLC – Clear Carry Flag

Description:

Clears the Carry Flag (C) in SREG (Status Register).

Operation:

(i) $C \leftarrow 0$

	Syntax:	Operands:	Program Counter:
(i)	CLC	None	PC ← PC + 1

16-bit Opcode:

1001	0100	1000	1000

Status Register (SREG) and Boolean Formula:

I	T	H	S	V	N	Z	C
–	–	–	–	–	–	–	0

Instructions with similar Syntax as CLC

- CLC
- SEC
- SEN
- CLN
- SEZ
- CLZ
- SEI
- CLI
- CLS
- SES
- SEV
- CLV
- SET
- CLT
- SEH
- CLH

COM – One's Complement

Description:

This instruction performs a One's Complement of register Rd.

Operation:

(i) Rd ← $FF - Rd

	Syntax:	Operands:	Program Counter:
(i)	COM Rd	$0 \leq d \leq 31$	PC ← PC + 1

16-bit Opcode:

1001	010d	dddd	0000

Status Register (SREG) and Boolean Formula:

I	T	H	S	V	N	Z	C
–	–	–	⇔	0	⇔	⇔	1

RET – Return from Subroutine

Description:

Returns from subroutine. The return address is loaded from the STACK. The Stack Pointer u
during RET.

Operation:

(i) PC(15:0) ← STACK Devices with 16 bits PC, 128K bytes Program memory maximum
(ii) PC(21:0) ← STACK Devices with 22 bits PC, 8M bytes Program memory maximum.

	Syntax:	Operands:	Program Counter:	Sta
(i)	RET	None	See Operation	SP
(ii)	RET	None	See Operation	SP

16-bit Opcode:

1001	0101	0000	1000

RJMP – Relative Jump

Description:

Relative jump to an address within PC - 2K +1 and PC + 2K (words). For AVR microcontrol
exceeding 4K words (8K bytes) this instruction can address the entire memory from every ad

Operation:

(i) PC ← PC + k + 1

	Syntax:	Operands:	Program Counter:	St
(i)	RJMP k	-2K ≤ k < 2K	PC ← PC + k + 1	Un

16-bit Opcode:

1100	kkkk	kkkk	kkkk

Example:

```
        cpi     r16,$42   ; Compare r16 to $42
        brne    error     ; Branch if r16 <> $42
        rjmp    ok        ; Unconditional branch
error:  add     r16,r17   ; Add r17 to r16
        inc     r16       ; Increment r16
ok:     nop               ; Destination for rjmp (do nothing)
```

SBIC – Skip if Bit in I/O Register is Cleared

Description:

This instruction tests a single bit in an I/O Register and skips the next instruction if the bit is
ates on the lower 32 I/O Registers – addresses 0-31.

Operation:

(i) If I/O(A,b) = 0 then PC ← PC + 2 (or 3) else PC ← PC + 1

	Syntax:	Operands:	Program Counter:
(i)	SBIC A,b	$0 \le A \le 31$, $0 \le b \le 7$	PC ← PC + 1, Condition fa PC ← PC + 2, Skip a one w PC ← PC + 3, Skip a two w

16-bit Opcode:

1001	1001	AAAA	Abbb

Example:

```
e2wait:  sbic  $1C,1      ; Skip next inst. if EEWE cleared
         rjmp  e2wait     ; EEPROM write not finished
         nop              ; Continue (do nothing)
```